Gottfried Wilhelm Leibniz Universität Hannover
Wirtschaftswissenschaftliche Fakultät
Institut für Wirtschaftsinformatik

Einsatz Künstlicher Intelligenz zur Erstellung von Energieprognosen – eine Analyse von KI Frameworks

Masterarbeit

zur Erlangung des akademischen Grades „Master of Science (M.Sc.)“ im Studiengang Wirtschaftswissenschaft der Wirtschaftswissenschaftlichen Fakultät der Leibniz Universität Hannover

vorgelegt von

Sven Kramer

[redacted]

Prüfer: Prof. Dr. Michael H. Breitner
Betreuer: M.Sc. Jana Gerlach

Hannover, den 30.09.2022

Inhaltsverzeichnis

Abbildungsverzeichnis .....
Tabellenverzeichnis .....
Formelverzeichnis .....
Abkürzungsverzeichnis .....
Research Summary .....
1. Einführung 1
1.1 Problemstellung 1
1.2 Zielsetzung, Motivation und Methodik 4
1.3 Aufbau 5
2. Theoretische Grundlagen 6
2.1 Definition der Künstlichen Intelligenz 6
2.2 Geschichte der KI 8
2.3. Machine Learning 10
2.3.1 Überwachtes Lernen 12
2.3.2 Unüberwachtes Lernen 13
2.3.3 Bestärkendes Lernen 14
2.3.4 Probleme von ML-Modellen 15
2.4 Künstliche Neuronale Netze und Deep Learning 16
2.4.1 Allgemeine Funktionsweise 17
2.4.2 Perzeptron 17
2.4.3 Mehrschichtige Neuronale Netze 18
2.4.4 Lernprozess 22
2.4.4.1 Verlustfunktion 22
2.4.4.2 Gradientenabstiegsverfahren 23
2.4.4.3 Backpropagation 27
2.4.4.4 AdaGrad-, RMS-Prop- und Adam-Optimierer 28
2.4.5 Über- und Unteranpassung 29
2.4.6 weitere Netztypen 30
2.5 Anwendungsbereiche von KI 30
2.5.1 generelle Anwendungsbereiche 30
2.5.2 KI in der Energiewirtschaft 32
2.5.2.1 Stromerzeugung 34
2.5.2.2 Stromvertrieb 36
2.5.2.3 Endnutzung von Strom 37
2.5.2.4 Stromnetze 39
3. verwendete Software-Technologien 40

3.1 verwendete Programmiersprache Python 41
3.2 verwendete KI-Frameworks 41
3.2.1 Scikit-learn.....41
3.2.2 Keras.....42
3.2.3 PyTorch.....42
4.Modellauswahl, Datenauswahl und Aufbereitung 43
4.1 Methodik und Modellauswahl.....43
4.2 Datenauswahl.....44
4.2.1 verwendete Outputdaten 44
4.2.2 verwendete Inputdaten 46
4.2.3 Datenanalyse der ausgewählten Parameter 48
4.3 Datenaufbereitung 51
4.3.1 Anpassung und Zusammenführung der Datensätze 51
4.3.2 Fehlerbereinigung und Normierung 53
5. Modellentwicklung der Frameworks.....54
5.1 Scikit-learn.....54
5.1.1 Implementierung.....54
5.1.2 Training und Validierung.....60
5.1.3 Optimierung.....62
5.2 Keras 66
5.2.1 Implementierung.....66
5.2.2 Training und Validierung.....67
5.2.3 Optimierung.....69
5.3 PyTorch 72
5.3.1 Implementierung.....72
5.3.2 Training und Validierung.....75
6. Analytischer Vergleich der Frameworks.....79
6.1 Prognosen und Ergebnisse.....79
6.2 Analyse und Diskussion der Ergebnisse 83
6.3 Analytischer Vergleich der Frameworks 85
6.3.1 Bedienbarkeit.....86
6.3.2 Dokumentation.....87
6.3.3 Servicequalität 89
6.3.4 Prognosequalität.....91
6.3.5 Abschließende Beurteilung 91
6.4 Limitationen der Arbeit 94
6.5 Erweiterungen und Handlungsempfehlungen.....96

7. Fazit 97
Literaturverzeichnis 102
Anhangsverzeichnis 114
Anhang.....115
Scikit-learn 120
load2019 data 121
Keras - Tensorflow 122
Pytorch.....123
Ehrenwörtliche Erklärung 127

Research Summary

I. Einleitung

Aufgrund stark steigender globaler Mitteltemperaturwerte hat die Problematik der globalen Klimaerwärmung vor allem in den letzten Jahrzehnten zunehmend an Relevanz gewonnen. Dabei steht außer Frage, dass der Klimawandel hauptsächlich durch den Anstieg an Treibhausgasemissionen verursacht wird (DWD, 2021). Um zahlreiche negative Folgeschäden verhindern zu können bedarf es demnach verschiedener Lösungsansätze zur Reduzierung dieser Emissionen. Vor allem der Ausbau regenerativer Energiequellen stellt hierbei einen wichtigen Faktor zur Bewältigung der Klimakrise dar (Sims, 2004). Da die Stromerzeugung dieser Energiequellen großen wetterbedingten Schwankungen unterliegt, sind zum Erhalt der Versorgungssicherheit genaue Prognosen der Stromerzeugung von großer Relevanz (Lange, 2011). In diesem Kontext können Anwendungen der Künstlichen Intelligenz einen entscheidenden Beitrag leisten. So ermöglichen Machine Learning-Prozesse eine erhöhte Prognosegenauigkeit in zahlreichen Anwendungsbereichen der Energiewirtschaft (Sehnke, et al., 2016). Da die Umsetzung dieser Prozesse durch KI-Frameworks erfolgt wird innerhalb dieser Arbeit auf dessen Anwendung eingegangen. So werden die populären KI-Frameworks Scikit-learn, Keras und PyTorch zur Implementierung von KNN-Modellen verwendet, woraufhin ein Leistungsvergleich stattfindet. Darüber hinaus sollen die Frameworks ebenfalls hinsichtlich ihrer Bedienbarkeit, Dokumentation, Servicequalität und Trainingsdauer miteinander verglichen werden. Während ein klassisches ML-Modell als Vergleich zu den KNNs herangezogen wird, erfolgt ebenfalls die Ermittlung eines optimalen Setups an Hyperparametern für erwähnte KNNs. Schließlich wird eine Nutzungsempfehlung für den betrachteten Anwendungsfall ausgesprochen.

II. Theoretische Grundlagen

A. Definition KI

Heutzutage existieren zahlreiche unterschiedliche Begriffsdefinitionen für Künstliche Intelligenz. sodass sich das Ableiten einer allgemeingültigen Definition als schwierig erweist. Dies liegt zum einen daran, dass sich das Forschungsfeld der KI über diverse verschiedene Anwendungsbereiche erstreckt. Zum anderen führte die rasante Weiterentwicklung von KI-Technologien dazu, dass Definitionen über den Entwicklungsverlauf stetig angepasst werden mussten (Wirtz & Weyerer, 2019). Erschwerend kommt hinzu, dass ohnehin keine allgemeingültige Definition für Intelligenz existiert (Mainzer, 2019). Ein ausgewählter integrativer Definitionsansatz fasst jedoch Aspekte verschiedener KI-Definitionen zusammen und beschreibt KI schließlich so: „AI beschreibt die Fähigkeit eines Computersystems menschenähnliches intelligentes Verhalten wiederzugeben, welches sich durch bestimmte Kernkompetenzen wie etwa Wahrnehmung, Verstehen, Handeln und Lernen auszeichnet, um menschliches und systemisches Verhalten bestmöglich zu unterstützen.“ (Wirtz, 2020, S. 261). Zudem wird heutzutage in der Literatur zwischen verschiedenen Ausprägungsformen von KI unterschieden, welche sich durch den Stärkegrad an intelligentem Verhalten klassifizieren

Hinsichtlich zusätzlicher Handlungsempfehlungen kann gesagt werden, dass sich die Wahl eines Frameworks auch an der Datenverteilung der Prognosewerte orientieren kann. Da die verschiedenen Modelle bei etwa gleicher Prognoseleistung eine unterschiedliche Verteilung ihrer Prognosewerte aufweisen, kann zur Bewältigung einer bestimmten Aufgabe jenes Modell herangezogen werden, dessen Verteilung der Prognosewerte am ehesten den Anforderungen der spezifischen Aufgabe entspricht.

7. Fazit

Innerhalb dieser Arbeit wurde die Anwendung verschiedener KI-Frameworks zur Prognose regenerativer Stromerzeugung betrachtet. Da für einen solchen Anwendungsfall verschiedene Frameworks zur Auswahl stehen, stellt sich die Frage, welches der verfügbaren Frameworks die beste Eignung aufweist. Während die Prognosequalität das relevanteste Auswahlkriterium darstellt, so sind auch die Kriterien der Bedienbarkeit, Dokumentation und Servicequalität von Interesse. Aufgrund dessen wurden die populären Frameworks Scikit-learn, Keras und PyTorch hinsichtlich dieser Kriterien miteinander verglichen. Hierzu wurde die Implementierung von KNN-Modellen ausführlich dokumentiert, sodass auch fachfremden Lesern der Einstieg in die Nutzung der Frameworks erleichtert werden kann. Jedoch auch nach Auswahl eines Frameworks stellt sich die Frage, welches Setup an Hyperparametern die beste Leistung hinsichtlich der Prognose regenerativer Stromerzeugung aufweist. So wurden für die erstellten KNNs der Frameworks mehrere Setups hinsichtlich ihrer Leistung miteinander verglichen, woraufhin das beste ausgewählt werden konnte.

Zu Beginn dieser Arbeit wurden zunächst sämtliche theoretischen Grundlagen der KI erläutert, woraufhin verschiedene Anwendungsbereiche innerhalb der Energiewirtschaft aufgezeigt wurden. Zur Umsetzung der Prognosen wurden folgend für den ostdeutschen Raum Stromerzeugungsdaten von Wind- und Solarkraftanlagen sowie Wetterdaten aufbereitet. Hierbei handelte es sich um frei verfügbare Datensätze die öffentlichen Plattformen entnommen wurden. Schließlich erfolgte die Modellimplementierung durch Verwendung der Frameworks SKL, Keras und PT. Nachdem für jedes Framework ein KNN-Modell erstellt wurde, erfolgte die Ermittlung eines optimalen Setups an Hyperparametern. Anschließend wurde die Leistung der Modelle anhand einer Prognose auf bisher ungesehene Testdaten evaluiert. Hierbei wurde eine Langzeitprognose angestrebt, weswegen die Stromerzeugungsdaten für den Zeitraum vom 01.01.2019 – 30.06.2019 prognostiziert wurden. Um einen Vergleich zu klassischen ML-Modellen herstellen zu können, wurden neben den Leistungsdaten der KNNs ebenfalls die des Gradient-Boosting-Regressors erfasst. Dieser wurde in SKL implementiert. Nach dem Vergleich der Prognosegenauigkeit wurden schließlich die verwendeten Frameworks hinsichtlich ihrer Bedienbarkeit, Dokumentation und Servicequalität miteinander verglichen, Zudem sollte ebenfalls die benötigte Trainingsdauer der Frameworks berücksichtigt werden, weswegen ein neues Kriterium erstellt wurde, welches die Prognosegenauigkeit und Trainingsdauer zusammenfasste.

Es stellte sich heraus, dass durch die Verwendung eines einzigen Hidden-Layers mit 32 Neuronen für jedes betrachtete KNN die beste Prognosegenauigkeit erzielt werden konnte.

Die Verwendung mehrerer Hidden-Layer und Neuronen pro Hidden-Layer führte vor allem bei der Prognose der W-Stromerzeugung zu einer Überanpassung des Modells. Die Einstellungen der weiteren Hyperparameter des optimalen Setups setzten sich folgendermaßen zusammen:

- ReLU-Aktivierungsfunktion, Adam-Optimierungsverfahren, Lernrate von 0.002, Batchgröße von 100, Epochenanzahl von 500.

Hierbei ist jedoch anzumerken, dass die Anpassung der Hyperparameter auf Basis der Prognose der S-Stromerzeugung erfolgte. Zwar liefert dieses Setup auch zur Prognose anderer Stromerzeugungsdaten eine gute Orientierung, jedoch sollte jedes Setup an die Datenstruktur der zu prognostizierenden Werte angepasst werden.

Bei Betrachtung der Prognosegenauigkeit auf die Testdaten aus dem Jahr 2019 fiel auf, dass die Leistungsunterschiede zwischen den KNNs sehr gering waren. Während sie hinsichtlich der Prognose der S-Stromerzeugung nahezu die gleiche Leistung erzielten, konnte das PT-Modell bei der Prognose der W-Stromerzeugung die besten Ergebnisse erzielen. Jedoch fiel auch diese Leistungsdifferenz so klein aus, dass sie aufgrund der Varianz der Ergebnisse als nicht signifikant betrachtet werden kann.

Zudem ist aufgefallen, dass obwohl die KNNs der verschiedenen Frameworks eine vergleichbare Leistung erzielten, sich die prognostizierten Werte in ihrer Datenstruktur unterschieden. Während der MLPR und das Keras-Modell die realen Stromerzeugungsdaten prinzipiell eher unterschätzten, so überschätzte das PT-Modell diese eher.

Anhand eines Vergleichs zwischen den KNNs und dem GBR konnte festgestellt werden, dass die KNNs die S-Stromerzeugung besser prognostizieren konnten. Dagegen konnte jedoch der GBR signifikant bessere Ergebnisse hinsichtlich der Prognose der W-Stromerzeugung erzielen.

Beim Vergleich der Prognoseleistung auf Validierungs- und Testdatensatz wurde festgestellt, dass bei der Prognose der S-Stromerzeugung die Leistungen der Modelle auf den Testdatensatz etwas schlechter ausfielen. Diese Leistungsdifferenzen bewegten sich jedoch in einem normalen Bereich. Da die Testdaten zeitlich weiter entfernt von den Trainingsdaten sind, war von einer etwas schlechteren Leistung auszugehen. Bezüglich der Prognose der W-Stromerzeugung fiel die Leistung der Modelle auf den Testdatensatz jedoch deutlich schlechter als auf den Validierungsdatensatz auf. Der Hauptgrund hierfür liegt aller Voraussicht nach darin, dass sich die Datenstruktur des Testdatensatzes stark von der des Trainings- und Validierungsdatensatzes unterscheidet.

SKL konnte in der Anwendung besonders durch seine einfache Bedienbarkeit positiv hervorstechen. Das Framework besitzt eine einfache Architektur, die Implementierung und Training von Modellen durch Verwendung weniger Codezeilen ermöglicht. Hierdurch wird ebenfalls die einfache Übertragung einer Aufgabenstellung auf verschiedene vorgefertigte ML-

Modelle ermöglicht, sodass schnell überprüft werden kann welches der verfügbaren Modelle die beste Eignung aufweist. Dazu bietet das Framework eine übersichtliche und verständliche Dokumentation aller Funktionen und Parameter. Neben der technischen Umsetzung der Modelle sind hier ebenfalls die theoretischen Grundlagen ausführlich dokumentiert. Der größte Nachteil von SKL besteht schließlich in den stark limitierten Konfigurationsmöglichkeiten der verfügbaren Modelle. So können lediglich simple FF-Netze mit der gleichen Aktivierungsfunktion innerhalb aller Hidden-Layer implementiert werden. Auch die Implementierung von Dropout-Layern oder einer Aktivierungsfunktion innerhalb des Output-Layers ist hier nicht möglich. Ein großer Nachteil besteht zudem darin, dass die Fehlerentwicklung über den Trainingsverlauf nicht visualisiert werden kann, was die Identifikationsmöglichkeiten von Überanpassung stark beschränkt. Trotz alledem konnte SKL mit Abstand die kürzeste Trainingsdauer vorweisen.

Auch Keras konnte durch eine relativ einfache Bedienbarkeit überzeugen. Dadurch, dass es mehrere TF-Funktionen vereinfacht zusammenfasst, kann es Komplexitäten von TF abstrahieren und benötigte Benutzeraktionen reduzieren. Für ein klassisches DL-Framework bietet es zugängliche und prägnante Codes, die ein schnelles Prototyping ermöglichen. Dazu kann Keras ebenfalls eine gute Dokumentation der verfügbaren Funktionen vorweisen. Anders als in SKL liegt hier der Fokus der Dokumentation jedoch ausschließlich auf der technischen Umsetzung der DL-Anwendungen und nicht auf deren theoretischen Grundlagen. Die Dokumentation von Keras profitiert besonders von der Integration in TF. So werden zahlreiche Keras-Anwendungen ebenfalls auf TF-Plattformen dokumentiert. Zur Implementierung von KNNs erlaubt Keras die Umsetzung der wichtigsten Konfigurationen. Alle zuvor erwähnten Modellkonfigurationen, die in SKL nicht möglich sind können in Keras umgesetzt werden. Darüber hinaus ermöglicht Keras die Visualisierung der Fehlerentwicklung über den Trainingsverlauf sowie die Ausführung von DL-Prozessen auf GPUs. Eine Einschränkung besteht darin, dass Keras ausschließlich die Umsetzung von DL-Anwendungen unterstützt. Zudem wies es unter allen betrachteten Frameworks die längste Trainingsdauer auf.

In PT ging die Bedienung mit einem erhöhten Komplexitätsgrad und Codierungsaufwand einher. Da es die objektorientierte Programmierung von Python verwendet, werden gewisse Vorkenntnisse vorausgesetzt. Für Implementierung, Training und Evaluierung des Modells wurden hier diverse zusätzliche Funktionen benötigt, welche in den anderen Frameworks optimiert zusammengefasst sind. Da PT mit Daten im Tensoren-Format arbeitet, musste zudem eine zusätzliche Vorbereitung der Daten vollzogen werden. Zudem ist die Dokumentation von PT etwas weniger umfangreich und übersichtlich als die der anderen Frameworks und bezieht sich nur auf technische Umsetzung von DL-Anwendungen. Da PT jedoch ein großes Nutzerwachstum aufweist, kann zukünftig ebenfalls von schnellen Verbesserungen in dieser Hinsicht ausgegangen werden. Der große Vorteil von PT besteht schließlich in den zahlreichen Konfigurationsmöglichkeiten der Modelle. Dabei ermöglicht es fast alle vorstellbaren Modellkonfigurationen, sodass es für ML-Prozesse aus allen Anwendungsbereichen verwendet werden kann. Hierbei können die verwendeten Modelle

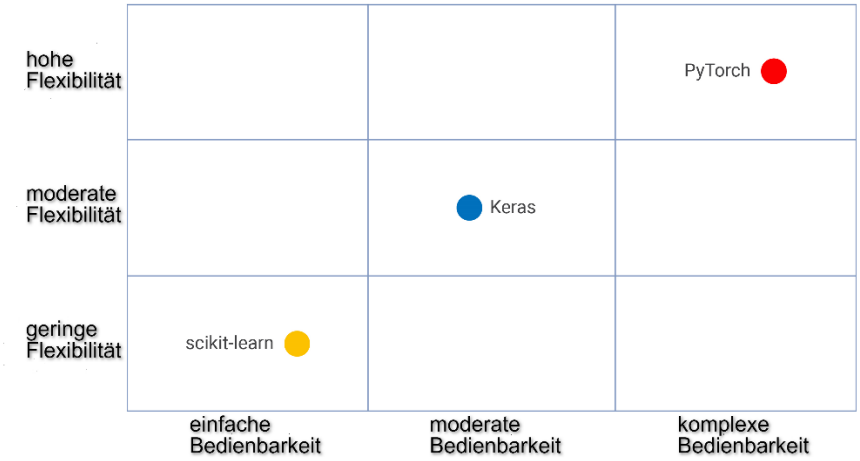
sehr spezifisch an den betrachteten Anwendungsfall angepasst werden. Hinsichtlich der Trainingsdauer bewegte sich PT zwischen den Frameworks SKL und Keras.

Bezogen auf den betrachteten Anwendungsfall der Prognose regenerativer Stromerzeugung stellt SKL schließlich eine sehr gute Wahl dar Es kombiniert die Aspekte einer einfachen Bedienbarkeit, übersichtlichen Dokumentation und kurzen Trainingsdauer mit einer Prognosegenauigkeit, die denen der anderen Frameworks ebenbürtig ist. Dadurch, dass SKL eine schnelle Implementierung verschiedener ML-Modelle ermöglicht, kann identifiziert werden welches der verfügbaren Modelle die beste Eignung für den betrachteten Anwendungsfall aufweist. Da der GBR hinsichtlich der Prognose der W-Stromerzeugung die beste Leistung aufweisen konnte, kann davon ausgegangen werden, dass für derartige Zeitreihenprognosen einer niedrigen Dimensionalität KNNs nicht immer die beste Eignung unter allen ML-Modellen aufweisen.

Sollte die genaue Identifikation von Überanpassung durch Visualisierung der Fehlerentwicklung von besonders großer Relevanz sein, so stellt Keras eine gute Wahl dar. Außerdem kann hier durch eine Implementierung von Dropout-Layern oder einer ReLU-Aktivierungsfunktion innerhalb der Outputschicht eine potentielle Verbesserung der Prognoseleistung erreicht werden.

Zwar weist PT die besten Konfigurationsmöglichkeiten der betrachteten Frameworks auf, jedoch scheinen die meisten hiervon nicht für Zeitreihenprognosen von Stromerzeugungsdaten benötigt zu werden. Außerdem besteht ein Trade-Off zwischen der einfachen Bedienbarkeit und hohen Flexibilität eines Frameworks. Abbildung 31 ordnet die betrachteten Frameworks schließlich hinsichtlich dieser beiden Kriterien ein:

Abbildung 31: Einordnung von SKL, Keras und PT hinsichtlich ihrer Bedienbarkeit und Flexibilität



Sollten in einem Anwendungsfall die Anforderungen an die Konfigurationsmöglichkeiten nicht allzu hoch sein, so kann durch die Wahl eines Frameworks mit einfacherer Bedienbarkeit

Aufwand vermieden werden. Anwendungsfälle die sehr komplexe Modelle erfordern benötigen jedoch diese zahlreichen Konfigurationsmöglichkeiten. Demnach sollte PT für rechenintensive Aufgaben einer hohen Dimensionalität, wie beispielsweise innerhalb der Sprach- und Bildverarbeitung, verwendet werden.